# A Region-Based Approach to Monocular Mapless Navigation Using Deep Learning Techniques

Machkour, Zakariae; Ortiz Arroyo, Daniel; Durdevic, Petar

*Published in:*
9th 2023 International Conference on Control, Decision and Information Technologies, CoDIT 2023

*DOI (link to publication from Publisher):*
10.1109/CoDIT58514.2023.10284463

*Publication date:*
2023

*Document Version*
Accepted author manuscript, peer reviewed version

# A Region-Based Approach to Monocular Mapless Navigation Using Deep Learning Techniques

Zakariae Machkour[1] and Daniel Ortiz-Arroyo[2] and Petar Durdevic[3]

***Abstract*— In recent years, there have been significant advances in navigation methods for autonomous robotic systems, giving rise to a diverse range of navigation techniques. These techniques include GPS-based, SLAM-based, and monocular depth-based navigation. However, each of these approaches has its limitations. Typically, these techniques rely on either external sensors and positioning systems or require the creation of a local map prior to initiating navigation. This paper introduces a new approach for autonomous navigation of ground robots: mapless navigation using a pre-trained monocular depth network. This technique offers an efficient and cost-effective way of navigating without the need for a pre-existing map of the environment. To evaluate and compare the performance of our method, we conducted experiments using two different depth estimation models tested within the Gazebo simulation environment.**

## I. INTRODUCTION

To navigate effectively, a robot must have the ability to perceive its environment with precision, devise a suitable path, and manipulate effectively its actuators. Typically, these abilities are achieved through depth measurements and object detection sensors, allowing the robot to recognize obstacles and determine their distances while navigating. These sensors are combined with actuators and a control system that governs the robot's speed and direction of movement.

Various proximity sensors, including LiDAR, RADAR, and ultrasonic sensors, have been employed to measure depth, enhancing the perception capabilities of robotic systems [1, 2, 3]. Compared to other depth sensors, monocular depth estimation provides a cost-effective solution for sensing depth. Unlike the previously mentioned alternatives, utilizing cameras (mono or stereo) for depth estimation offers a more affordable approach [4]. Although stereo cameras can provide the required depth information [5], they are frequently orders of magnitude more expensive than a monocular camera [6].

Irrespective of the sensors employed, navigation systems can be categorized into three distinct groups: map-based, map-building, and map-less systems. Surveys on navigation systems for robots describing map-based and mapless methods were presented in [7, 8, 9]. Map-based navigation systems depend on geometric models or topological maps of the environment [10]. In this case, the robot has a model of its surroundings and can detect collisions by calculating distances. Map-building systems can build maps of the environment during runtime and localize the robot position relative to the created map, typically using the Simultaneous Localization and Mapping (SLAM) [11, 12] technology. However, current visual SLAM (vSLAM) technology is often unstable and prone to error, particularly in complex dynamic environments that change over time [13, 14, 15]. The limitations of vSLAM in such environments include issues like error accumulation, illumination changes, fast motion, dynamic obstacles, and the assumption of static features in the environment. While deep learning techniques can enhance object recognition accuracy in complex scenes when combined with vSLAM, their real-time performance is often insufficient, limiting their practical applications [13, 15]. Both map-based and map-building approaches frequently rely on path planning, using deterministic discretization techniques or probabilistic-based algorithms [16]. These algorithms generate feasible paths in 2D or 3D for efficient exploration and collision checking in the mapped environment using deterministic or probabilistic methods.

Conversely, mapless-based systems function in the absence of an environment map and instead utilize sensor data to perceive the environment and generate motion commands for robot navigation. Unlike map-based and map-building systems, mapless approaches remove the cost of storing and maintaining a map, allowing robots with constrained computing power to operate in arbitrarily large environments. Furthermore, it is impractical to build and maintain a map when the robot operates for a short period and has only one chance to navigate an unknown, dynamic, and complex environment [17].

The goal of this paper is to investigate the effective utilization of monocular depth estimation (MDE) for robot navigation in indoor environments that differ from the model's training settings. The study aims to address the challenges associated with the accuracy limitations of the depth map. Specifically, we exploit the relative positions of objects in a scene, as opposed to relying solely on using accurate depth estimates for navigation. Additionally, we take advantage of segmentation techniques to partition the depth map into regions, allowing the robot to look at the scene in a more intuitive and structured manner.

## II. RELATED WORKS

In [18], a mapless navigation technique using landmark images was used to route the robot to a target place. The system employed landmark names and images as input to a CNN, which produces a bounding box. In the Visual Servoing (VS) system used, the difference between the centers of the bounding box and the input image was utilized to

*This work was not supported by any organization

[1,2,3]Authors are with the Department of Energy, Aalborg University, Niels Bohrs Vej 8, 6700 Esbjerg, Denmark

[1,2,3]{zma, doa, pdl}@energy.aau.dk

compute the direction toward the target point. A second DNN collected the original and desired images from the target location, outputting the required movements to reach the target position. However, the proposed VS method presents several issues:

1) The navigation system does not consider the dynamic environment (moving obstacles), whether is cluttered or sparse.
2) The system learns the relationship between the initial and reference image, which means that a new network has to be trained for each reference pose, making it impractical for new environments.
3) The system is limited to the presence of landmarks.

In a more recent study conducted by Zhang et al. [19], LiDAR data was employed as visual features in a Visual Servoing (VS) system for positioning tasks. Their proposed VS system leverages the pose relationship between the desired and current point clouds to guide the robot's movement from an initial position to the desired destination.

Likewise, Tsai et al. [20] employed a mapless LiDAR navigation control approach for a wheeled mobile robot. The method proposed by the authors utilizes a deep CNN network to learn the correlation between the input LiDAR data and the desired motion behavior, utilizing end-to-end imitation learning. A motion prediction module predicts the motion behavior of the robot. The success rate reported was 75% on average.

In another similar study by Nguyen et al. [21], an autonomous navigation solution called NMFNet is proposed for a mobile wheeled robot. The authors utilize an RGB-D camera to capture RGB images and point clouds, while a Lidar device is used to construct a distance map.

NMFNet comprises three modules designed to learn depth features from the distance map, extract features from RGB images, and process 3D point cloud data. The outputs of these three modules are fused to predict the steering angle, enabling the robot to navigate autonomously.

In a study by Xiong et al. [22], another Visual Servoing (VS) approach utilizing RGB-D data for robot navigation toward a target location is presented. The method proposed drives the robot by minimizing the depth map error between the initial and target positions.

Additionally, several recent research papers have conducted surveys on the current advancements in vision-based robot navigation and visual servoing. Examples include Li et al. [23], Islam et al. [24], Xiaomotion [25], and Machkour et al. [26].

Lastly, another approach to implementing mapless navigation is to use deep reinforcement learning techniques. An example of this method is [27] where navigation policies are learned using the actor-critic algorithm and a special reward function that prioritizes curiosity-driven exploration.

It is important to note that previous examples cover various aspects of vision-based robot navigation but none of them uses monocular depth estimation (MDE), in their visual servoing navigation systems. Our system was evaluated within an indoor environment. Furthermore, our approach

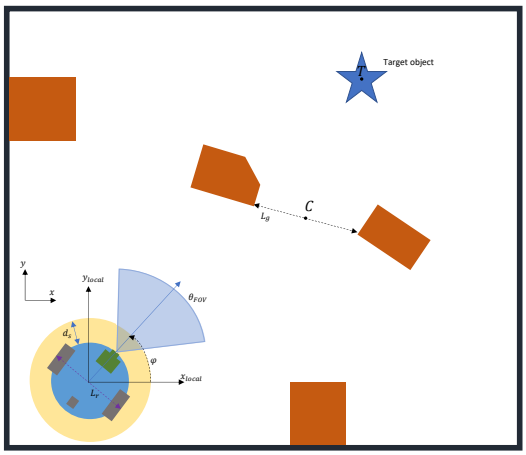


Fig. 1. Obstacle avoidance problem definition.

uses the two-step navigation method described in reference [18]. While the method in [18] incorporates landmark images and uses a CNN for bounding box detection and visual servoing, our approach identifies gaps within the camera's field of view and utilizes a MDE model for segmentation of the colored map. This segmentation information guides our robot's movement toward the target object while avoiding obstacles.

Our contribution, in this work is the design of a novel mapless obstacle avoidance algorithm that exploits monocular depth map estimation. The MDE network feeds the estimated signal to a PD controller. Our navigation system integrates object detection networks for object tracking and obstacle avoidance, with the MDE providing depth information for obstacle avoidance. Our approach differs from previous works by solely relying on a pre-trained MDE model for obstacle avoidance and segmenting the depth map into regions to use them in obstacle detection. Finally, we evaluate and compare the performance of our method with different depth estimation models, namely BTS [28] and MiDaS [29]. BTS is a network architecture, that employs local planar guidance layers to establish a direct connection between internal feature maps and the desired prediction. This approach enhances the network's training process, resulting in improved performance. MiDaS implements a flexible loss function and a dataset mixing strategy to improve performance.

## III. Problem Formulation

In our previous work [30], we considered a point-mass robot in a bounded closed space $\mathcal{W} \subset \mathbb{R}^3$ which consists of $n \in \mathbb{N}$ obstacles $\mathcal{O}_i \subset \mathcal{W}$, $i \in 1, ..., n$. We denote $\mathcal{F}$ as the free space in $\mathcal{W}$ as $\mathcal{F} = \mathcal{W} \setminus \bigcup_{i=1}^{n} \mathcal{O}_i$.

We utilize a two-wheel nonholonomic differential drive robot equipped with a monocular camera in our research. The robot's goal is to reach a target object in the environment without relying on a map (see Fig. 2). Instead, it relies solely on the visual information captured by the monocular camera for navigation. By leveraging the robot's differential drive

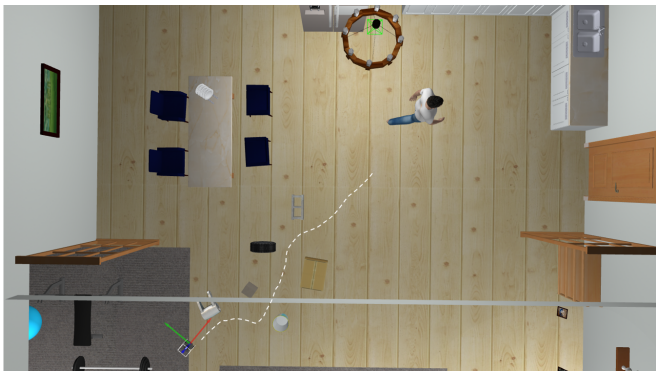

Fig. 2. The expected successful path towards the target object.

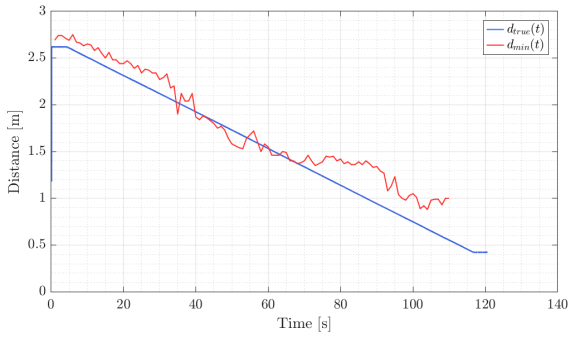


Fig. 3. Estimated distance inaccuracy.

capabilities and the visual input from the camera, we aim to enable the robot to navigate towards the target object effectively, without the need for accessing a map.

The robot must detect the free spaces or gaps within its camera's Field Of View (FOV), and choose the appropriate gap through which can navigate without collision. We define $L_g$ as the gap length between two obstacles, and $L_r$ as the robot length (see Fig. 1 and 4). The navigation in the workspace $\mathcal{W}$ is plausible if:

$$\exists \varphi \in [0, 2\pi], \exists \mathcal{O}_i, \mathcal{O}_j \subset \mathcal{W}, i, j \in 1, ..., n \mid n \in \mathbb{N}^*, i \neq j$$
$$\forall p, q \in \mathcal{O}_i \times \mathcal{O}_j, L_g > L_r, L_g = \min \|\mathbf{p} - \mathbf{q}\| \quad (1)$$

with $\varphi$ being the rotation angle of the robot around the z-axis. In the implementation, $p$ and $q$ represent the points of intersection between the circumferences of the obstacles, denoted as $\mathcal{O}_i$ and $\mathcal{O}_j$, respectively, and the line that defines the safety distance, $d_s$. These points of intersection are obtained by determining where the circumferences of the obstacles intersect with the line representing the minimum allowable distance from the robot.

## IV. Proposed Method

To navigate successfully towards a target object using our proposed method, the robot must be able to :

1) Detect the obstacles inside a given distance range.
2) Detect the gaps between these obstacles.
3) Choose the appropriate gap and navigate through it.
4) Detect and navigate toward the target object.

We address the aforementioned problems using mainly a monocular camera as the robot sensor.

### A. Obstacle detection

Obstacle avoidance cannot be accomplished solely with object detection models due to the challenge of learning about all potential objects that may be present in an unknown scene. Instead, we rely on a Monocular Depth Estimation (MDE) model [29] to detect unknown obstacles. The depth map image used in our approach is estimated with a Deep Neural Network (DNN) rather than obtained from an active sensor such as Lidar or Kinect. As a result, the distance measurements derived from the estimated depth map may exhibit inaccuracies in real-world scenarios. In our previous work [30], we compared the estimated distance using the pre-trained MDE model BTS [28] with the true distance as a moving robot approaches a target object and it showed big disparities between the estimated and actual distances (see Fig. 3).

We propose to use the relative positions of objects in the scene to overcome the requirement of having a precise depth map. First, we define a safety distance $d_s$ (see Fig.4) within which we need to keep out all obstacles while navigating. This safe space can also be considered as the ground space, which can be tagged in the depth map as a close space to the robot. The idea is to ignore everything in this space and consider only the obstacles outside it.

To detect the obstacles within the robot camera's FOV, we generate the monocular depth map and apply a threshold on it based on the distance color related to the accepted range. This will allow the algorithm to track only the objects that are within a certain distance range and eliminate all other objects from the depth map.

### B. Free space detection

In this stage, we generate a binary map from the thresholded depth map (see Fig. 4) to detect the obstacles and the possible gaps between them. We assign 0 to free spaces and 1 to obstacles. In order to localize the gaps between the obstacles, we take the horizontal line defining the top edge of the safe space and we extract the corresponding array of pixel values (see Fig. 5). Each group of 1s denotes an obstacle $\mathcal{O}_i$, and each group of 0s is a gap $G_j$ with a length $L_{G_j}$.

### C. Best gap selection

For plausible navigation, we need to find a free space $G_j$ with $L_{G_j} > L_r$ (see Eqn. (1)). However, $L_{G_j}$ is expressed in pixels and should be converted to real-world coordinates. The two extremities of the gap $G_j$ are defined as $(x_1, y_1)$ and $(x_2, y_2)$, with $y_1 = y_2 = d_s$. By knowing the camera's intrinsic parameters and utilizing the depth map generated by a Monocular depth estimation model, we can have this

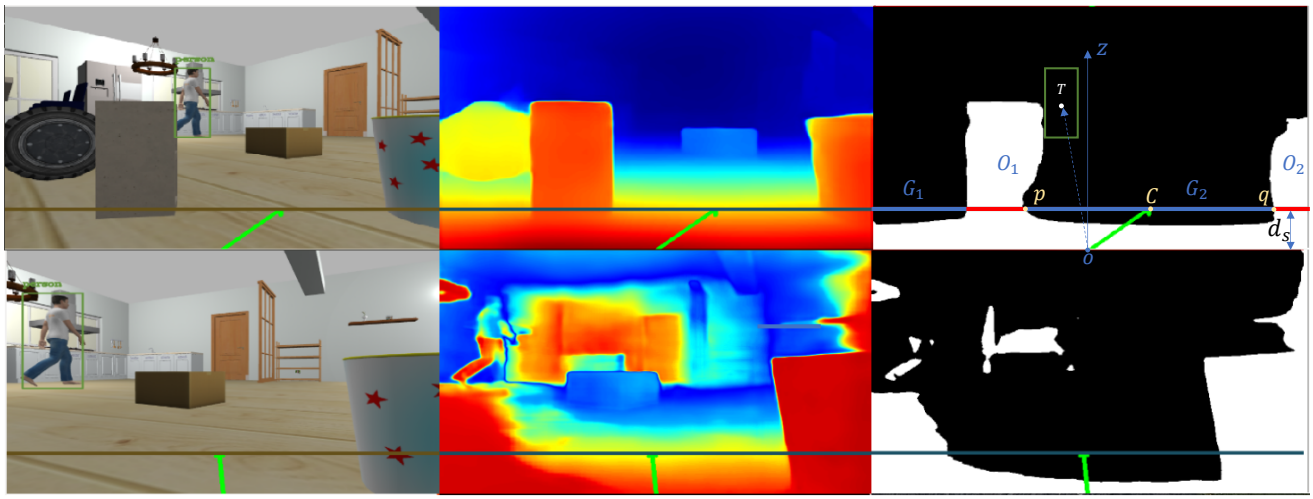


Fig. 4. Robot navigation. From left to right: input image, colored depth map, binary map. Top: MiDaS-based navigation. Bottom: BTS-based navigation. $d_s$ is the safety distance for robotic navigation. The green arrow points to the best gap.

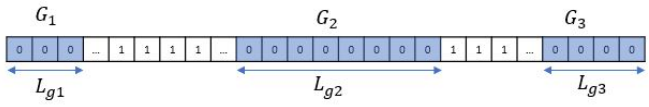


Fig. 5. Gaps and obstacles detection.

conversion as follows:

$$\begin{aligned} X_1 &= Z_1 \frac{x_1}{f} \\ Y_1 &= Z_1 \frac{y_1}{f} \\ X_2 &= Z_2 \frac{x_2}{f} \\ Y_2 &= Z_2 \frac{y_2}{f} \end{aligned} \tag{2}$$

with $Z_1$ and $Z_2$ are the distances towards the point $(x_1, y_1)$ and the point $(x_2, y_2)$, respectively, and $f$ is the camera's focal length. The converted value of $L_{Gj}$ becomes:

$$L_{Gj} = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \tag{3}$$

We define the set of candidate gaps as :

$$S_G = \{G_j, j \in 1, ..., n \in \mathbb{N}^* \mid L_{G_j} > L_r\} \tag{4}$$

and among this set $S_G$, we choose the most appropriate gap $G_a \in S_G$, with $C_a$ as the center of $G_a$ and $\phi_a = \angle(\mathbf{OC_a}, \mathbf{OT})$ the bearing angle between $OC_a$ and $OT$ the center of the target object, such as:

$$\begin{aligned} &\forall \phi_i = \angle(\mathbf{OC_i}, \mathbf{OT}), |\phi_a| = \min_{G_i \in S_G} |\phi_i|, \\ &\text{if target object within FOV.} \\ &L_{G_a} = \max_{G_i \in S_G} L_{G_i}, \text{otherwise.} \end{aligned} \tag{5}$$

In our experiments, the target is a person that is visible to the robot from the beginning of the experiment. The robot is programmed to autonomously navigate toward the target using our proposed method, which identifies gaps within the camera's field of view. When the target is not visible to the robot, a fail-safe mechanism makes the robot switch to an exploring mode, where its primary objective becomes obstacle avoidance.

### *D. Navigate towards the target object*

The PD controller regulates the robot's yaw (angular) and linear speed based on the information extracted from the monocular depth estimation (MDE) model. Using MDE's output the controller determines the appropriate yaw angle to steer the robot toward the desired direction by determining gaps. The yaw error ($\epsilon_{\text{ygap}}$) is calculated by subtracting half of the image width ($\frac{w_{\text{img}}}{2}$) from the gap center ($c_{\text{gap}}$). Based on the sign of the resulting value, $\epsilon_{\text{ygap}}$ is recalculated for the best performance during navigation:

$$\epsilon_{\text{ygap}} = c_{\text{gap}} - \frac{w_{\text{img}}}{2} \tag{6}$$

$$\epsilon_{\text{ygap}} = \begin{cases} 2 \cdot c_{\text{gap}} \cdot 0.25 - \frac{w_{\text{img}}}{2}, & \text{if } \epsilon_{\text{ygap}} < 0 \\ 2 \cdot c_{\text{gap}} \cdot 0.75 - \frac{w_{\text{img}}}{2}, & \text{otherwise} \end{cases} \tag{7}$$

The values $0.25$ and $0.75$ in the equation 7 serve to determine the appropriate position for pointing towards the gaps in the left and right sides, respectively. When the gap is located on the left side of the image, the goal is to direct the attention slightly towards the edge of the gap rather than the center. To achieve this, we calculate the value $2 * c_{\text{gap}} * 0.25$, which represents a position 25% into the gap length from its left edge. Similarly, when the gap is on the right side, we calculate the value $2 * c_{\text{gap}} * 0.75$, which corresponds to a position 25% into the gap length from its right edge. These values help adjust the direction of focus towards the desired portion of the gap, enhancing the navigation and obstacle avoidance strategy. We also calculate the yaw error ($\epsilon_{\text{ytarget}}$) by subtracting half of the image width ($\frac{w_{\text{img}}}{2}$) from the target value ($c_{\text{target}}$):

$$\epsilon_{\text{ytarget}} = c_{\text{target}} - \frac{w_{\text{img}}}{2} \tag{8}$$

The following equation ensures obstacle avoidance while keeping track of the target. To prioritize obstacle avoidance, we assign a higher weight or impact $\beta$ to the gap component.

$$\epsilon_{\text{combined}} = \beta \cdot \epsilon_{\text{ygap}} + (1 - \beta) \cdot \epsilon_{\text{ytarget}} \tag{9}$$

The yaw speed $\omega_{\text{yaw}}$ of the robot is controlled using a PID controller. The control equation is given by:

$$\omega_{\text{yaw}} = -\left(k_p \cdot \epsilon_{\text{combined},t} + k_d \cdot (\epsilon_{\text{combined},t} - \epsilon_{\text{combined},t-1})\right) \tag{10}$$

In this equation, $k_p$ and $k_d$ are the controller parameters representing the proportional and derivative gains, respectively. The variable $\epsilon_{\text{combined},t}$ represents the combined yaw error between two frames at time step $t$, while $\epsilon_{\text{combined},t-1}$ represents the combined yaw error at the previous time step. The value of $\omega_{\text{yaw}}$ is clipped to limit it within a specified range. While varying the yaw speed, we maintain a constant linear speed throughout the experiment. We employ a PD controller with $Ki = 0$, $Kp = Kd = 0.5$ in the navigation algorithm for the turtlebot3. The P and D gains were found experimentally. The proportional and derivative terms perform the robot's yaw control, allowing precise maneuvering.

## V. Results and Discussion

We conducted multiple experiments to evaluate the performance of our method. Figure 6 illustrates the trajectories of the robot using 1) our method with the MiDaS model, 2) our method with the BTS model, and 3) the turtlebot3_navigation ROS package. The obstacles in the figure are represented by their geometric centers for illustrative purposes. The trajectory obtained with the MiDaS-based navigation showed the best results, successfully making the robot navigate through obstacles following a shorter path compared to the ros package and the BTS-based navigation.

Table I shows that, on average, the robot exhibits more successful navigations towards the target object when the MiDaS model is employed, as compared to the BTS model. This result can be attributed to the fact that BTS was trained only on the NYU dataset that consists of 2.8 GB of specific types of indoor video scenes captured by a handheld RGB and Kinect Microsoft camera. Additionally in [30], we evaluated the BTS model with varying camera heights and found that when the camera was mounted at 20 cm height on the TurtleBot3 robot, the depth accuracy was low. This is illustrated in Figure 4, which provides a comparison between the BTS model (top) and MiDaS (bottom). The MiDaS model incorporates a loss function that included scale- and shift-invariant loss together with multiscale and a scale-invariant gradient matching term adapted to the disparity space. Additionally, the MiDaS model was trained using diverse measurement tools and with multiple datasets. Lastly, the model was tested with other unseen datasets. Consequently, MiDaS exhibits state-of-the-art performance

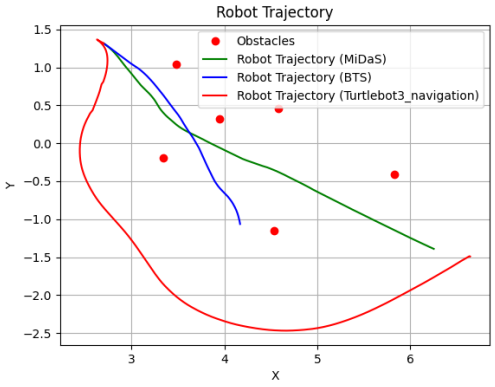


Fig. 6. Comparison of Robot Trajectories: MiDaS vs. BTS vs. turtlebot3_navigation ROS package.

in depth estimation and greater adaptability to unknown scenes compared to the BTS model.

| MDE model | #Success | #Failure | Success rate |
|---|---|---|---|
| BTS | 17 | 13 | 56% |
| MiDaS | 28 | 2 | 93% |

TABLE I
Success Rates of MDE-Based Obstacle Avoidance Models

In table I, the column header "MDE Model" indicates the results corresponding to the BTS and MiDaS models. "#Success" is the number of successful navigation instances where the robot was able to reach the target location without colliding. "#Failure" is the number of unsuccessful navigation instances where the robot collided with an obstacle resulting in failure to reach the target location.

## VI. Conclusion

Our proposed method demonstrates the benefit of using multiple deep learning models for monocular mapless navigation. The integration of object detection and monocular depth estimation networks, along with region-based segmentation, enables the designing of a robust and efficient obstacle avoidance and object tracking system. Further, our findings indicate that the performance of our mapless navigation algorithm significantly depends on the accuracy of the depth estimator model used. As part of our future work, we plan to evaluate the system under diverse testing scenarios with multiple robots, humans, and varying lighting conditions to further assess its performance.

## References


[1] César Debeunne and Damien Vivet. "A review of visual-LiDAR fusion based simultaneous localization and mapping". In: *Sensors* 20.7 (2020), p. 2068.

[2] Sarah H Cen and Paul Newman. "Precise ego-motion estimation with millimeter-wave radar under diverse and challenging conditions". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 6045–6052.

[3] Joon Hyo Rhee and Jiwon Seo. "Low-cost curb detection and localization system using multiple ultrasonic sensors". In: *Sensors* 19.6 (2019), p. 1389.

[4] Muhammad Abdul Haseeb et al. "DisNet: a novel method for distance estimation from monocular camera". In: *10th Planning, Perception and Navigation for Intelligent Vehicles (PPNIV18), IROS* (2018).

[5] Petar Durdevic and Daniel Ortiz-Arroyo. "A deep neural network sensor for visual servoing in 3d spaces". In: *Sensors* 20.5 (2020), p. 1437.

[6] Kyle J Cantrell, Craig D Miller, and Carlos Morato. "Practical Depth Estimation with Image Segmentation and Serial U-Nets." In: *VEHITS*. 2020, pp. 406–414.

[7] Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. "Visual navigation for mobile robots: A survey". In: *Journal of intelligent and robotic systems* 53 (2008), pp. 263–296.

[8] Guilherme N DeSouza and Avinash C Kak. "Vision for mobile robot navigation: A survey". In: *IEEE transactions on pattern analysis and machine intelligence* 24.2 (2002), pp. 237–267.

[9] Francisco Rubio, Francisco Valero, and Carlos Llopis-Albert. "A review of mobile robots: Concepts, methods, theoretical framework, and applications". In: *International Journal of Advanced Robotic Systems* 16.2 (2019), p. 1729881419839596.

[10] B Madhevan and M Sreekumar. "Identification of probabilistic approaches and map-based navigation in motion planning for mobile robots". In: *Sādhanā* 43.1 (2018), pp. 1–18.

[11] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM". In: *European conference on computer vision*. Springer. 2014, pp. 834–849.

[12] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE transactions on robotics* 31.5 (2015), pp. 1147–1163.

[13] Jun Cheng et al. "A review of visual SLAM methods for autonomous driving vehicles". In: *Engineering Applications of Artificial Intelligence* 114 (2022), p. 104992.

[14] Yong-bao Ai et al. "Visual SLAM in dynamic environments based on object detection". In: *Defence Technology* 17.5 (2021), pp. 1712–1721.

[15] Zewen Xu, Zheng Rong, and Yihong Wu. "A survey: which features are required for dynamic visual simultaneous localization and mapping?" In: *Visual Computing for Industry, Biomedicine, and Art* 4.1 (2021), pp. 1–16.

[16] Daniel Ortiz-Arroyo. "A hybrid 3D path planning method for UAVs". In: *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*. IEEE. 2015, pp. 123–132.

[17] Fred de Villiers and Willie Brink. "Learning fine-grained control for mapless navigation". In: *2020 International SAUPEC/RobMech/PRASA Conference*. IEEE. 2020, pp. 1–6.

[18] Hiroki Kanayama et al. "Two-mode mapless visual navigation of indoor autonomous mobile robot using deep convolutional neural network". In: *2020 IEEE/SICE International Symposium on System Integration (SII)*. IEEE. 2020, pp. 536–541.

[19] Shiyi Zhang et al. "A Visual Servoing Method based on Point Cloud". In: *2020 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE. 2020, pp. 369–374.

[20] Chi-Yi Tsai, Humaira Nisar, and Yu-Chen Hu. "Mapless LiDAR Navigation Control of Wheeled Mobile Robots Based on Deep Imitation Learning". In: *IEEE Access* 9 (2021), pp. 117527–117541.

[21] Anh Nguyen and Quang D Tran. "Autonomous navigation with mobile robots using deep learning and the robot operating system". In: *Robot Operating System (ROS)*. Springer, 2021, pp. 177–195.

[22] Yufeng Xiong et al. "3D depth map based optimal motion control for wheeled mobile robot". In: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2017, pp. 2045–2050.

[23] Chenping Li et al. "A survey on visual servoing for wheeled mobile robots". In: *International Journal of Intelligent Robotics and Applications* 5.2 (2021), pp. 203–218.

[24] Shafiqul Islam, Jorge Dias, and Anderson Sunda-Meya. "On the Design and Development of Vision-Based Autonomous Mobile Manipulation". In: *IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society*. IEEE. 2021, pp. 1–6.

[25] Xuesu Xiao et al. "Motion Planning and Control for Mobile Robot Navigation Using Machine Learning: a Survey". In: ().

[26] Zakariae Machkour, Daniel Ortiz-Arroyo, and Petar Durdevic. "Classical and Deep Learning based Visual Servoing Systems: a Survey on State of the Art". In: *Journal of Intelligent & Robotic Systems* 104.1 (2022), pp. 1–27.

[27] Oleksii Zhelo et al. *Curiosity-driven Exploration for Mapless Navigation with Deep Reinforcement Learning*. 2018. arXiv: 1804.00456 [cs.RO].

[28] Jin Han Lee et al. "From big to small: Multi-scale local planar guidance for monocular depth estimation". In: *arXiv preprint arXiv:1907.10326* (2019).

[29] René Ranftl et al. "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer". In: *IEEE transactions on pattern analysis and machine intelligence* 44.3 (2020), pp. 1623–1637.

[30] Zakariae Machkour, Daniel Ortiz-Arroyo, and Petar Durdevic. "Monocular Based Navigation System for Autonomous Ground Robots Using Multiple Deep Learning Models". In: *International Journal of Computational Intelligence Systems* 16.1 (2023), p. 79.